

# Hankel Matrices for Weighted Visibly Pushdown Automata

Nadia Labai<sup>\*1</sup> and Johann A. Makowsky<sup>†2</sup>

<sup>1</sup>Department of Informatics, Vienna University of Technology [labai@forsyte.at](mailto:labai@forsyte.at)

<sup>2</sup>Department of Computer Science, Technion - Israel Institute of Technology  
[janos@cs.technion.ac.il](mailto:janos@cs.technion.ac.il)

## Abstract

Hankel matrices (aka connection matrices) of word functions and graph parameters have wide applications in automata theory, graph theory, and machine learning. We give a characterization of real-valued functions on nested words recognized by weighted visibly pushdown automata in terms of Hankel matrices on nested words. This complements C. Mathissen’s characterization in terms of weighted monadic second order logic.

## 1 Introduction and Background

### 1.1 Weighted Automata for Words and Nested Words

Classical word automata can be extended to *weighted* word automata by assigning weights from some numeric domain to their transitions, thereby having them assign values to their input words rather than accepting or rejecting them. Weighted (word) automata define the class of recognizable word functions, first introduced in the study of stochastic automata by A. Heller [39]. Weighted automata are used in verification, [6, 52], in program synthesis, [13, 14], in digital image compression, [19], and speech processing, [53, 28, 1]. For a comprehensive survey, see the Handbook of Weighted Automata [27]. Recognizable word functions over commutative semirings  $\mathcal{S}$  were characterized using logic through the formalism of Weighted Monadic Second Order Logic (WMSOL), [26], and the formalism of MSOLEVAL<sup>1</sup>, [44].

Nested words and nested word automata are generalizations of words and finite automata, introduced by Alur and Madhusudan [3]. A *nested word*  $nw \in \text{NW}(\Sigma)$  over an alphabet  $\Sigma$  is a sequence of linearly ordered positions, augmented with forward-oriented edges that do not cross, creating a nested structure. In the context of formal verification for software, execution paths in procedural programs are naturally modeled by nested words whose hierarchical structure captures calls and returns. Nested words also model annotated linguistic data and tree-structured data which is given by a linear encoding, such as HTML/XML documents. Nested word automata define the class of regular languages of nested words. The key feature of these automata is their ability to propagate hierarchical states along the augmenting edges, in addition to the

---

<sup>\*</sup>Supported by the National Research Network RiSE (S114), and the LogiCS doctoral program (W1255) funded by the Austrian Science Fund (FWF).

<sup>†</sup>Partially supported by a grant of Technion Research Authority.

<sup>1</sup>This formalism was originally introduced in [18] for graph parameters.

states propagated along the edges of the linear order. We refer the reader to [3] for details. Nested words  $nw \in \text{NW}(\Sigma)$  can be (linearly) encoded as words over an extended tagged alphabet  $\hat{\Sigma}$ , where the letters in  $\hat{\Sigma}$  specify whether the position is a call, a return, or neither (internal). Such encodings of regular languages of nested words give the class of *visibly pushdown languages* over the tagged alphabet  $\hat{\Sigma}$ , which lies between the parenthesis languages and deterministic context-free languages. The accepting pushdown automata for visibly pushdown languages push one symbol when reading a call, pop one symbol when reading a return, and only update their control when reading an internal symbol. Such automata are called *visibly pushdown automata*. Since their introduction, nested words and their automata have found applications in specifications for program analysis [24, 37, 25], XML processing [31, 54], and have motivated several theoretical questions, [20, 2, 55].

Visibly pushdown automata and nested word automata were extended by assigning weights from a commutative semiring  $\mathcal{S}$  to their transitions as well. Kiefer et al introduced *weighted visibly pushdown automata*, and their equivalence problem was showed to be logspace reducible to polynomial identity testing, [41]. Mathissen introduced *weighted nested word automata*, and proved a logical characterization of their functions using a modification of WMSOL, [51].

## 1.2 Hankel Matrices and Weighted Word Automata

Given a word function  $f : \Sigma^* \rightarrow \mathcal{F}$ , its *Hankel matrix*  $\mathbf{H}_f \in \mathcal{F}^{\Sigma^* \times \Sigma^*}$  is the infinite matrix whose rows and columns are indexed by words in  $\Sigma^*$  and  $\mathbf{H}_f(u, v) = f(uv)$ , where  $uv$  is the concatenation of  $u$  and  $v$ . In addition to the logical characterizations, there exists a characterization of recognizable word functions via Hankel matrices, by Carlyle and Paz [12].

**Theorem 1** (Carlyle and Paz, 1971). *A real-valued word function  $f$  is recognized by a weighted (word) automaton iff  $\mathbf{H}_f$  has finite rank.*

The theorem was originally stated using the notion of external function rank, but the above formulation is equivalent. Multiplicative words functions were characterized by Cobham [15] as exactly those with a Hankel matrix of rank 1.

Hankel matrices proved useful also in the study of graph parameters. Lovász introduced a kind of Hankel matrices for graph parameters [48] which were used to study real-valued graph parameters and their relation to partition functions, [30, 49]. In [33], the definability of graph parameters in monadic second order logic was related to the rank of their Hankel matrices. Meta-theorems involving logic, such as Courcelle’s theorem and generalizations thereof [23, 16, 17, 50], were made logic-free by replacing their definability conditions with conditions on Hankel matrices, [45, 43, 46].

## 1.3 Our Contribution

The goal of this paper is to prove a characterization of the functions recognizable by weighted visibly pushdown automata (WVPA), called here *recognizable nested word functions*, via Hankel matrices. Such a characterization would nicely fill the role of the Carlyle-Paz theorem in the words setting, complementing results that draw parallels between recognizable word functions and nested word functions, such as the attractive properties of closure and decidability the settings share [3], and the similarity between the WMSOL-type formalisms used to give their logical characterizations.

The first challenge is in the choice of the Hankel matrices at hand. A naive straightforward adaptation of the Carlyle-Paz theorem to the setting of nested words would involve Hankel matrices for words over the extended alphabet  $\hat{\Sigma}$  with the usual concatenation operation on words. However, then we would have functions recognizable by WVPA with Hankel matrices of infinite

rank. Consider the Hankel matrix of the characteristic function of the language of balanced brackets, also known as the Dyck language. This language is not regular, so its characteristic function is not recognized by a weighted word automaton. Hence, by the [Carlyle-Paz theorem](#), its Hankel matrix would have infinite rank despite the fact its encoding over a tagged alphabet is recognizable by VPA, hence also by WVPA.

## Main results

We introduce *nested Hankel matrices* over *well-nested words* (see [section 2](#)) to overcome the point described above and prove the following characterization of WVPA-recognizable functions of well-nested words:

**Theorem 2** (Main Theorem).

*Let  $\mathcal{F} = \mathbb{R}$  or  $\mathcal{F} = \mathbb{C}$ , and let  $f$  be an  $\mathcal{F}$ -valued function on well-nested words. Then  $f$  is recognized by a weighted visibly pushdown automaton with  $n$  states iff the nested Hankel matrix  $\mathbf{nH}_f$  has rank  $\leq n^2$ .*

As opposed to the characterizations of word functions, which allow  $f$  to have values over a semiring, we require that  $f$  is over  $\mathbb{R}$  or  $\mathbb{C}$ . This is due to the second challenge, which stems from the fact that in our setting of functions of well-nested words, the helpful decomposition properties exhibited by Hankel matrices for word functions are absent. This is because, as opposed to words, well-nested words cannot be split in arbitrary positions and result in two well-nested words. Thus, we use the [singular value decomposition \(SVD\) theorem](#), see, e.g., [\[35\]](#), which is valid only over  $\mathbb{R}$  and  $\mathbb{C}$ .

## Outline

In [section 2](#) we complete the background on well-nested words and weighted visibly pushdown automata, and introduce nested Hankel matrices. The rather technical proof of [Theorem 2](#) is given in [section 5](#). In [section 3](#) we discuss the applications of [Theorem 2](#) to learning theory. In [section 4](#) we briefly discuss limitations of our methods and possible extensions of our characterization.

## 2 Preliminaries

For the remainder of the paper, we assume that  $\mathcal{F}$  is  $\mathbb{R}$  or  $\mathbb{C}$ . Let  $\Sigma$  be a finite alphabet. For  $\ell \in \mathbb{N}^+$ , we denote the set  $\{1, \dots, \ell\}$  by  $[\ell]$ . For a matrix or vector  $\mathbf{N}$ , denote its transpose by  $\mathbf{N}^T$ . Vectors are assumed to be column vectors unless stated otherwise.

### 2.1 Well-Nested Words

We follow the definitions in [\[3\]](#) and [\[51\]](#). A *well-nested word* over  $\Sigma$  is a pair  $(w, \nu)$  where  $w \in \Sigma^*$  of length  $\ell$  and  $\nu$  is a matching relation for  $w$ . A *matching relation*<sup>2</sup> for a word of length  $\ell$  is a set of edges  $\nu \subset [\ell] \times [\ell]$  such that the following holds:

1. If  $(i, j) \in \nu$ , then  $i < j$ .
2. Any position appears in an edge of  $\nu$  at most once: For  $1 \leq i \leq \ell$ ,  $|\{j \mid (i, j) \in \nu\}| \leq 1$  and  $|\{j \mid (j, i) \in \nu\}| \leq 1$

---

<sup>2</sup>The original definition of nested words allowed “dangling” edges. We will only be concerned with nested words that are well-matched.

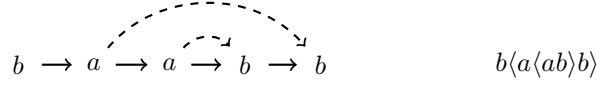


Figure 1: On the left, a well-nested word, where the successor relation of the linear order is in bold edges, the matching relation is in dashed edges. On the right, its encoding as a word over a tagged alphabet.

3. If  $(i, j), (i', j') \in \nu$ , then it is not the case that  $i < i' \leq j < j'$ . That is, the edges do not cross.

Denote the set of well-nested words over  $\Sigma$  by  $\text{WNW}(\Sigma)$ .

Given positions  $i, j$  such that  $(i, j) \in \nu$ , position  $i$  is a *call* position and position  $j$  is a *return* position. Denote  $\Sigma_{\text{call}} = \{\langle s \mid s \in \Sigma \rangle\}$ ,  $\Sigma_{\text{ret}} = \{s \mid s \in \Sigma\}$ , and  $\hat{\Sigma} = \Sigma_{\text{call}} \cup \Sigma_{\text{ret}} \cup \Sigma_{\text{int}}$  where  $\Sigma_{\text{int}} = \Sigma$  and is disjoint from  $\Sigma_{\text{call}}$  and  $\Sigma_{\text{ret}}$ . By viewing calls as opening parentheses and returns as closing parentheses, one can define an encoding taking nested words over  $\Sigma$  to words over  $\hat{\Sigma}$  by assigning to a position labeled  $s \in \Sigma$ :

- ◆ the letter  $\langle s$ , if it is a call position,
- ◆ the letter  $s \rangle$ , if it is a return position,
- ◆ the same letter  $s$ , if it is an internal position.

We denote this encoding by  $nw\_w : \text{WNW}(\Sigma) \rightarrow \hat{\Sigma}^*$  and give an example in [Figure 1](#). Note that any parentheses appearing in such an encoding will be well-matched (balanced) parentheses. Denote its partial inverse function, defined only for words with well-matched parentheses, by  $w\_nw : \hat{\Sigma}^* \rightarrow \text{WNW}(\Sigma)$ . See [\[3\]](#) for details. We will freely pass between the two forms.

Given a function  $f : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$  on well-nested words, one can naturally define a corresponding function  $f' : \hat{\Sigma}^* \rightarrow \mathcal{F}$  on words with well-matched parentheses by setting  $f'(w) = f(w\_nw(w))$ . We will denote both functions by  $f$ .

## 2.2 Nested Hankel Matrices

Given a function on well-nested words  $f : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$ , define its *nested Hankel matrix*  $\mathbf{nH}_f$  as the infinite matrix whose rows and columns are indexed by *words* over  $\hat{\Sigma}$  with *well-matched parentheses*, and  $\mathbf{nH}_f(u, v) = f(uv)$ . That is, the entry at the row labeled with  $u$  and the column labeled with  $v$  is the value  $f(uv)$ . A nested Hankel matrix  $\mathbf{nH}_f$  has finite rank if there is a finite set of rows in  $\mathbf{nH}_f$  that linearly span it. We stress the fact that  $\mathbf{nH}_f$  is defined over words whose parentheses are well-matched, as this is crucial for the proof of [Theorem 2](#).

As an example, consider the function  $f$  which counts the number of pairs of parentheses in a well-nested word over the alphabet  $\Sigma = \{a\}$ . Then the corresponding word function is on words over the tagged alphabet  $\hat{\Sigma} = \{a, \langle a, a \rangle\}$ . In [Figure 2](#) we see (part of) the corresponding nested Hankel matrix  $\mathbf{nH}_f$  with labels on its columns and rows.

## 2.3 Weighted Visibly Pushdown Automata

For notational convenience, now let  $\Sigma = \Sigma_{\text{call}} \cup \Sigma_{\text{ret}} \cup \Sigma_{\text{int}}$ . We follow the definition given in [\[41\]](#).

	$\varepsilon$	$a$	$\langle aa \rangle$	$aa$	$\langle aaa \rangle$	$\langle a\langle aa \rangle a \rangle$	$\dots$
$\varepsilon$	0	0	1	0	1	2	$\dots$
$a$	0	0	1	0	1	2	$\dots$
$\langle aa \rangle$	1	1	2	1	2	3	$\dots$
$aa$	0	0	1	0	1	2	$\dots$
$\langle aaa \rangle$	1	1	2	1	2	3	$\dots$
$\langle a\langle aa \rangle a \rangle$	2	2	3	2	3	4	$\dots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	

Figure 2: The nested Hankel matrix  $\mathbf{nH}_f$ . Note that  $\mathbf{nH}_f$  has rank 2.

**Definition 3** (Weighted visibly pushdown automata). *An  $\mathcal{F}$ -weighted VPA on  $\Sigma$  is a tuple  $A = (n, \alpha, \eta, \Gamma, M)$  where*

- ◆  $n \in \mathbb{N}^+$  is the number of states,
- ◆  $\alpha, \eta \in \mathcal{F}^n$  are initial and final vectors, respectively,
- ◆  $\Gamma$  is a finite stack alphabet, and
- ◆  $M$  are matrices in  $\mathcal{F}^{n \times n}$  defined as follows.

For every  $\gamma \in \Gamma$  and every  $c \in \Sigma_{\text{call}}$ , the matrix  $\mathbf{M}_{\text{call}}^{(c, \gamma)} \in \mathcal{F}^{n \times n}$  is given by

$\mathbf{M}_{\text{call}}^{(c, \gamma)}(i, j)$  = the weight of a  $c$ -labeled transition from  
state  $i$  to state  $j$  that pushes  $\gamma$  onto the stack.

The matrices  $\mathbf{M}_{\text{ret}}^{(r, \gamma)} \in \mathcal{F}^{n \times n}$  are given similarly for every  $r \in \Sigma_{\text{ret}}$ , and the matrices  $\mathbf{M}_{\text{int}}^{(s)} \in \mathcal{F}^{n \times n}$  are given similarly for every  $s \in \Sigma_{\text{int}}$ .

**Definition 4** (Behavior of weighted VPA). *Let  $A = (n, \alpha, \eta, \Gamma, M)$  be an  $\mathcal{F}$ -weighted VPA on  $\Sigma$ . For a well-nested word  $u \in \text{WNW}(\Sigma)$ , the automaton  $A$  inductively computes a matrix  $\mathbf{M}_u^{(A)} \in \mathcal{F}^{n \times n}$  for  $u$  in the following way.*

- ◆ *Base cases:*

$$\mathbf{M}_\varepsilon^{(A)} = \mathbf{I}, \quad \text{and} \quad \mathbf{M}_s^{(A)} = \mathbf{M}_{\text{int}}^{(s)} \quad \text{for } s \in \Sigma_{\text{int}}.$$

- ◆ *Closure:*

$$\begin{aligned} \mathbf{M}_{uv}^{(A)} &= \mathbf{M}_u^{(A)} \cdot \mathbf{M}_v^{(A)} && \text{for } u, v \in \text{WNW}(\Sigma), \text{ and} \\ \mathbf{M}_{\text{cur}}^{(A)} &= \sum_{\gamma \in \Gamma} \mathbf{M}_{\text{call}}^{(c, \gamma)} \cdot \mathbf{M}_u^{(A)} \cdot \mathbf{M}_{\text{ret}}^{(r, \gamma)} && \text{for } c \in \Sigma_{\text{call}} \text{ and } r \in \Sigma_{\text{ret}}. \end{aligned}$$

The behavior of  $A$  is the function  $f_A : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$  where

$$f_A(u) = \alpha^T \cdot \mathbf{M}_u^{(A)} \cdot \eta$$

A function  $f : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$  is recognizable by weighted VPA if it is the behavior of some weighted VPA  $A$ .

### 3 Applications in Computational Learning Theory

A passive learning algorithm for classical automata is an algorithm which is given a set of strings accepted by the target automaton (positive examples) and a set of strings rejected by the target automaton (negative examples), and is required to output an automaton which is consistent with the set of examples. It is well known that in a variety of passive learning models, such as Valiant’s PAC model, [58], and the mistake bound models of Littlestone and Haussler et al, [47, 38], it is intractable to learn or even approximate classical automata, [34, 4, 56]. However, the problem becomes tractable when the learner is allowed to make membership and equivalence queries, as in the active model of learning introduced by Angluin, [4, 5]. This approach was extended to weighted automata over fields, [10].

The problem of learning *weighted* automata is of finding a weighted automaton which closely estimates some target function, by considering examples consisting of pairs of strings with their value. The development of efficient learning techniques for weighted automata was immensely motivated by the abundance of their applications, with many of the techniques exploiting the relationship between weighted automata and their Hankel matrices, [9, 36, 11].

#### 3.1 Learning Weighted Visibly Pushdown Automata

The proof of our [Theorem 2](#) suggests a template of learning algorithms for weighted visibly pushdown automata, with the difficult part being the construction of the matrices that correspond to call and return symbols. The proof of [Lemma 11](#) spells out the construction of these matrices, given an algorithm for finding SVD expansions (see [section 6](#)) and a spanning set of the nested Hankel matrix. To the best of our knowledge, learning algorithms for weighted visibly pushdown automata have not been proposed so far.

In recent years, the spectral method of Hsu et al [40] for learning hidden Markov models, which relies on the SVD of a Hankel matrix, has driven much follow-up research, see the survey [8]. Balle and Mohri combined spectral methods with constrained matrix completion algorithms to learn arbitrary weighted automata, [7]. We believe the possibility of developing spectral learning algorithms for WVPA is worth exploring in more detail.

Lastly, we should note that one could employ existing algorithms to produce a weighted automaton from a nested Hankel matrix, if it is viewed as a partial Hankel matrix for a word function. However, any automaton which is consistent with the matrix will have as many states as the rank of the nested Hankel matrix, [12, 29]. This may be less than satisfying when considering how, in contrast, [Theorem 2](#) assures the existence of a weighted visibly pushdown automaton with  $n$  states, given a nested Hankel matrix of rank  $\leq n^2$ . This discrepancy fundamentally depends on the [SVD Theorem](#).

### 4 Extension to Semirings

The proof of [Theorem 2](#) relies on the [SVD theorem](#), which, in particular, assumes the existence of an inverse with respect to addition. Furthermore, notions of orthogonality, rank, and norms do not readily transfer to the semiring setting. Thus it is not clear what an analogue to the SVD theorem would be in the context of semirings, nor whether it could exist. Therefore the proof of [Theorem 2](#) cannot be used to characterize nested word functions recognized by WVPA over semirings.

However, in the special case of the tropical semirings, De Schutter and De Moor proposed an extended max algebra corresponding to  $\mathbb{R}$ , called the *symmetrized max algebra*, and proved an analogue SVD theorem for it, [21]. See also [22] for an extended presentation. These results

suggest a similar Hankel matrix based characterization for WVPA-recognizable nested word functions may be possible over the tropical semirings. This would be beneficial in situations where we have a function that has a nested Hankel matrix of infinite rank when interpreted over  $\mathbb{R}$ , but has finite rank when it is interpreted over a tropical semiring. It is easy to verify that any function on well-nested words which is maximizing or minimizing with respect to concatenation would fall in this category.

## 5 The Characterization of WVPA-Recognizability

In this section we prove both directions of [Theorem 2](#).

### 5.1 Recognizability Implies Finite Rank of Nested Hankel Matrix

This is the easier direction of [Theorem 2](#). First we need a definition. For  $\ell, m \in [n]$ , define the matrix  $\mathbf{A}^{(\ell, m)} \in \mathcal{F}^{n \times n}$  as having the value 1 in the entry  $(\ell, m)$  and zero in all other entries. That is,

$$\mathbf{A}^{(\ell, m)}(i, j) = \begin{cases} 1, & \text{if } (i, j) = (\ell, m) \\ 0, & \text{otherwise} \end{cases}$$

Obviously, for any matrix  $\mathbf{M} \in \mathcal{F}^{n \times n}$  with entries  $\mathbf{M}(i, j) = m_{ij}$  we have  $\mathbf{M} = \sum_{i, j \in [n]} m_{ij} \mathbf{A}^{(i, j)}$ .

**Theorem 5.** *Let  $f : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$  be recognized by a weighted visibly pushdown automaton  $A$  with  $n$  states. Then the nested Hankel matrix  $\mathbf{nH}_f$  has rank  $\leq n^2$ .*

*Proof.* We describe infinite row vectors  $\mathbf{v}^{(i, j)}$  where  $i, j \in [n]$ , whose entries are indexed by well-nested words  $w \in \text{WNW}(\Sigma)$ , and show they span the rows of  $\mathbf{nH}_f$ . We define the entry of  $\mathbf{v}^{(i, j)}$  associated with  $w$  to be

$$\mathbf{v}^{(i, j)}(w) = \boldsymbol{\alpha}^T \cdot \mathbf{A}^{(i, j)} \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta}$$

Note that there are  $n^2$  such vectors. Now let  $u \in \text{WNW}(\Sigma)$  be a well-nested word and let  $\mathbf{M}_u^{(A)}$  be the matrix computed for  $u$  by  $A$  as described in the . Then the row  $\mathbf{r}_u$  corresponding to  $u$  in  $\mathbf{nH}_f$  has entries

$$\mathbf{r}_u(w) = \boldsymbol{\alpha}^T \cdot \mathbf{M}_u^{(A)} \cdot \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta}$$

We show this row is linearly spanned by the vectors  $\mathbf{v}^{(i, j)}$ ,  $i, j \in [n]$ . Consider the linear combination

$$\mathbf{v}_u = \sum_{1 \leq i, j \leq n} \mathbf{M}_u^{(A)}(i, j) \cdot \mathbf{v}^{(i, j)}.$$

Then

$$\begin{aligned} \mathbf{v}_u(w) &= \sum_{1 \leq i, j \leq n} \mathbf{M}_u^{(A)}(i, j) \cdot \mathbf{v}^{(i, j)}(w) = \sum_{1 \leq i, j \leq n} \mathbf{M}_u^{(A)}(i, j) \cdot \left( \boldsymbol{\alpha}^T \cdot \mathbf{A}^{(i, j)} \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta} \right) \\ &= \boldsymbol{\alpha}^T \cdot \mathbf{M}_u^{(A)} \cdot \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta} = \mathbf{r}_u(w) \end{aligned}$$

Therefore the rank of  $\mathbf{nH}_f$  is at most  $n^2$ . □

## 6 Finite Rank of Nested Hankel Matrix Implies Recognizability

Here we prove the second direction of [Theorem 2](#). This will be done by defining a weighted VPA which recognizes the function  $f$  described by a given Hankel matrix  $\mathbf{nH}_f$ . It will hold that if the rank of  $\mathbf{nH}_f$  is  $\leq n^2$ , the automaton will have at most  $n$  states, and a stack alphabet  $\Gamma$  of size at most  $n$ .

We first describe the initial and final vectors  $\boldsymbol{\alpha}, \boldsymbol{\eta}$  and the matrices that will be used to construct the automaton, and prove useful properties for them.

As we cannot decompose the Hankel matrix entries in arbitrary positions, but only in ways that maintain the well-nesting, we will need to use the following theorem to show the matrices used in the construction of the automaton indeed exist:

**Theorem 6** (The SVD Theorem, see [\[35\]](#)). *Let  $\mathbf{N} \in \mathcal{F}^{m \times n}$  be a non-zero matrix, where  $\mathcal{F} = \mathbb{R}$  or  $\mathcal{F} = \mathbb{C}$ . Then there exist vectors  $\mathbf{x}_1, \dots, \mathbf{x}_m \in \mathcal{F}^m$  and  $\mathbf{y}_1, \dots, \mathbf{y}_n \in \mathcal{F}^n$  such that the matrices*

$$\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_m] \in \mathcal{F}^{m \times m}, \quad \mathbf{Y} = [\mathbf{y}_1 \dots \mathbf{y}_n] \in \mathcal{F}^{n \times n}$$

are orthogonal, and

$$\mathbf{Y}^T \mathbf{N} \mathbf{X} = \text{diag}(\sigma_1, \dots, \sigma_p) \in \mathcal{F}^{m \times n}$$

where  $p = \min\{m, n\}$ ,  $\text{diag}(\sigma_1, \dots, \sigma_p)$  is a diagonal matrix with the values  $\sigma_1, \dots, \sigma_p$  on its diagonal, and  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p$ .

As a consequence, if we define  $r$  by  $\sigma_1 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = 0$ , that is the number of non-zero entries in  $\text{diag}(\sigma_1, \dots, \sigma_p)$ , then we have the SVD expansion of  $\mathbf{N}$ :

$$\mathbf{N} = \sum_{i=1}^r \sigma_i \mathbf{x}_i \mathbf{y}_i^T$$

In particular, if  $\mathbf{N}$  is of rank 1, then  $\mathbf{N} = \mathbf{xy}^T$ .

The SVD is perhaps the most important factorization for real and complex matrices. It is used in matrix approximation techniques, signal processing, computational statistics, and many more areas. See [\[42, 57, 32\]](#) and references therein.

### 6.0.1 The Components of the Automaton

Let  $\Sigma = \Sigma_{\text{call}} \cup \Sigma_{\text{ret}} \cup \Sigma_{\text{int}}$ , where  $\Sigma_{\text{call}}, \Sigma_{\text{ret}}$  and  $\Sigma_{\text{int}}$  are disjoint. Throughout this section, let  $f : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$  be a function on well-nested words over  $\Sigma$ , and let its nested Hankel matrix  $\mathbf{nH}_f$  have finite rank  $r(\mathbf{nH}_f) \leq n^2$ . Denote by  $\mathcal{B} = \{w_{1,1}, \dots, w_{n,n}\}$  the well-nested words whose rows linearly span  $\mathbf{nH}_f$ .

**Definition 7** (Initial and final vectors). *Let the matrix  $\mathbf{N} \in \mathcal{F}^{n \times n}$  be defined as  $\mathbf{N}(i, j) = f(w_{1,j})$ , and let  $\mathbf{x}, \mathbf{y} \in \mathcal{F}^n$  be vectors such that  $\mathbf{N} = \mathbf{xy}^T$ . Let*

$$\boldsymbol{\eta} = \mathbf{y}, \quad \boldsymbol{\alpha} = \mathbf{x} \tag{1}$$

Note that this definition is sound; as  $\mathbf{N}$  has rank 1, [Theorem 6](#) guarantees there exist such vectors  $\mathbf{x}, \mathbf{y}$ .



**Definition 8** (Internal matrices). For  $w_{i,j} \in \mathcal{B}$ , define  $\beta_{i,j} = f(w_{i,j})f(w_{1,j})^{-1}$ , and let

$$\mathbf{M}_{w_{i,j}} = \beta_{i,j} \cdot \mathbf{A}^{(i,j)} \quad (2)$$

Note that for  $w_{1,j}$ , we have  $\beta_{1,j} = 1$  and  $\mathbf{M}_{w_{1,j}} = \mathbf{A}^{(1,j)}$ .

For a letter  $a \in \Sigma_{int}$ , let  $\mathbf{r}_a$  denote the row in  $\mathbf{nH}_f$  corresponding to  $a$ . The rows indexed by the elements in  $\mathcal{B}$  span the matrix, so there is a linear combination of them equal to  $\mathbf{r}_a$ :

$$\mathbf{r}_a = \sum_{1 \leq i,j \leq n} z_a^{i,j} \cdot \mathbf{r}_{w_{i,j}}$$

Set

$$\mathbf{M}_a = \sum_{1 \leq i,j \leq n} z_a^{i,j} \cdot \mathbf{M}_{w_{i,j}} \quad (3)$$

Before we define the call and return matrices, we show the above defined vectors and matrices behave as expected:

**Lemma 9.** Let  $\alpha, \eta \in \mathcal{F}^n$ ,  $\mathbf{M}_{w_{i,j}} \in \mathcal{F}^{n \times n}$  for  $w_{i,j} \in \mathcal{B}$ , and  $\mathbf{M}_a$  for  $a \in \Sigma_{int}$  be defined as in Definitions 7 and 8. It holds that:

$$f(w_{i,j}) = \alpha^T \cdot \mathbf{M}_{w_{i,j}} \cdot \eta \quad (4)$$

$$f(a) = \alpha^T \cdot \mathbf{M}_a \cdot \eta \quad (5)$$

*Proof.* Since the entries of  $\mathbf{M}_{w_{i,j}}$  are zero except for entry  $(i,j)$ , we have

$$\alpha^T \cdot \mathbf{M}_{w_{i,j}} \cdot \eta = \alpha(i) \cdot \beta_{i,j} \cdot \eta(j)$$

Since  $\alpha(i)\eta(j) = f(w_{1,j})$ , we have

$$\alpha^T \cdot \mathbf{M}_{w_{i,j}} \cdot \eta = \beta_{i,j} \cdot f(w_{1,j}) = f(w_{i,j}) \cdot f(w_{1,j})^{-1} \cdot f(w_{1,j}) = f(w_{i,j})$$

and Equation 4 holds.

By the definition of  $\mathbf{M}_a$ , we have

$$\begin{aligned} \alpha^T \cdot \mathbf{M}_a \cdot \eta &= \alpha^T \left( \sum_{1 \leq i,j \leq n} z_a^{i,j} \cdot \mathbf{M}_{w_{i,j}} \right) \eta = \sum_{1 \leq i,j \leq n} z_a^{i,j} (\alpha^T \mathbf{M}_{w_{i,j}} \eta) \\ &= \sum_{1 \leq i,j \leq n} z_a^{i,j} \cdot f(w_{i,j}) = f(a) \end{aligned}$$

and Equation 5 holds.  $\square$

**Definition 10** (Call and return matrices). For each pair  $c \in \Sigma_{call}$  and  $r \in \Sigma_{ret}$ , define the  $n \times n$  matrix  $\mathbf{N}_{c,r}$  as  $\mathbf{N}_{c,r}(i,j) = f(cw_{i,j}r)/\beta_{i,j}$  and let its SVD be

$$\mathbf{N}_{c,r} = \sum_{k=1}^n \mathbf{p}_{c,k} (\mathbf{p}_{r,k})^T$$

For  $\gamma \in \Gamma$ , define

$$\mathbf{M}_{call}^{c,\gamma}(\ell, i) = \begin{cases} \mathbf{p}_{c,\gamma}(i)/\alpha(\gamma) & \ell = \gamma \\ 0 & \text{else} \end{cases}$$

and

$$\mathbf{M}_{ret}^{r,\gamma}(j, m) = \begin{cases} \mathbf{p}_{r,\gamma}(j)/\eta(\gamma) & m = \gamma \\ 0 & \text{else} \end{cases}$$

Now we show these matrices behave as expected:

**Lemma 11.** *Let  $\mathbf{M}_{call}^{(c,\gamma)}$  and  $\mathbf{M}_{ret}^{(r,\gamma)}$  for  $c \in \Sigma_{call}$ ,  $r \in \Sigma_{ret}$  be defined as in Definition 10. Then it holds that:*

$$f(cw_{i,j}r) = \boldsymbol{\alpha}^T \left( \sum_{\gamma=1}^n \mathbf{M}_{call}^{(c,\gamma)} \cdot \mathbf{M}_{w_{i,j}} \cdot \mathbf{M}_{ret}^{(r,\gamma)} \right) \boldsymbol{\eta} \quad (6)$$

*Proof.* By the definition of  $\mathbf{N}_{c,r}$  we have:

$$\mathbf{N}_{c,r}(i, j) = f(cw_{i,j}r) / \beta_{i,j} = \sum_{k=1}^n \mathbf{p}_{c,k}(i) \mathbf{p}_{r,k}(j) \quad (7)$$

In addition,  $\mathbf{M}_{w_{i,j}}$  is zero in all entries that are not the  $(i, j)$  one. Therefore,

$$(\mathbf{M}_{call}^{c,\gamma} \cdot \mathbf{M}_{w_{i,j}})(\ell, m) = \begin{cases} \mathbf{M}_{call}^{c,\gamma}(\gamma, i) \beta_{i,j} = \mathbf{p}_{c,\gamma}(i) \beta_{i,j} / \boldsymbol{\alpha}(\gamma) & \ell = \gamma, m = j \\ 0 & \text{else} \end{cases}$$

Thus multiplying with  $\mathbf{M}_{ret}^{r,\gamma}$  results in:

$$(\mathbf{M}_{call}^{c,\gamma} \cdot \mathbf{M}_{w_{i,j}} \cdot \mathbf{M}_{ret}^{r,\gamma})(\ell, m) = \begin{cases} (\mathbf{p}_{c,\gamma}(i) \beta_{i,j} \mathbf{p}_{r,\gamma}(j)) / (\boldsymbol{\alpha}(\gamma) \boldsymbol{\eta}(\gamma)) & \ell = m = \gamma \\ 0 & \text{else} \end{cases}$$

Note that the above matrix  $\mathbf{M}_{call}^{c,\gamma} \cdot \mathbf{M}_{w_{i,j}} \cdot \mathbf{M}_{ret}^{r,\gamma}$  is diagonal. Therefore, in total:

$$\begin{aligned} \boldsymbol{\alpha}^T \left( \sum_{\gamma=1}^n \mathbf{M}_{call}^{c,\gamma} \mathbf{M}_{w_{i,j}} \mathbf{M}_{ret}^{r,\gamma} \right) \boldsymbol{\eta} &= \beta_{i,j} \sum_{\gamma=1}^n \mathbf{p}_{c,\gamma}(i) \mathbf{p}_{r,\gamma}(j) \\ &\stackrel{\text{Equation 7}}{=} \beta_{i,j} (f(cw_{i,j}r) / \beta_{i,j}) = f(cw_{i,j}r) \end{aligned}$$

so Equation 6 holds.  $\square$

## 6.0.2 Putting the Automaton Together

We are now ready to prove the second direction of Theorem 2:

**Theorem 12.** *Let  $f : \text{WNW}(\Sigma) \rightarrow \mathcal{F}$  have a nested Hankel matrix  $\mathbf{nH}_f$  of rank  $\leq n^2$ . Then  $f$  is recognizable by a weighted visibly pushdown automaton  $A$  with  $n$  states.*

*Proof.* Use Definitions 7, 8, and 10 to build a weighted VPA  $A$  with  $n$  states, and set  $\mathbf{M}_\varepsilon^{(A)} = \mathbf{I}$ . From Lemmas 9 and 11 it only remains to show that for  $u, t \in \text{WNW}(\Sigma)$ ,

$$\mathbf{M}_{ut}^{(A)} = \mathbf{M}_u^{(A)} \cdot \mathbf{M}_t^{(A)}$$

Note that we defined the matrices  $\mathbf{M}_{w_{i,j}}^{(A)}$  such that  $\mathbf{r}_{w_{i,j}} = \mathbf{v}^{(i,j)}$  up to a constant factor. We show that if

$$\mathbf{r}_u = \sum_{1 \leq i, j \leq n} \mathbf{M}_u^{(A)}(i, j) \cdot \mathbf{v}^{(i,j)} \quad \text{and} \quad \mathbf{r}_t = \sum_{1 \leq i, j \leq n} \mathbf{M}_t^{(A)}(i, j) \cdot \mathbf{v}^{(i,j)},$$

then

$$\mathbf{r}_{ut} = \sum_{1 \leq i, j \leq n} (\mathbf{M}_u^{(A)} \cdot \mathbf{M}_t^{(A)})(i, j) \cdot \mathbf{v}^{(i,j)}$$

Or, equivalently, that for every well-nested word  $w \in \text{WNW}(\Sigma)$ ,

$$\mathbf{r}_{ut}(w) = \boldsymbol{\alpha}^T \cdot \mathbf{M}_u^{(A)} \cdot \mathbf{M}_t^{(A)} \cdot \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta}$$

Consider the linear combination:

$$\mathbf{v}_{ut} = \sum_{1 \leq i, j \leq n} (\mathbf{M}_u^{(A)} \cdot \mathbf{M}_t^{(A)})(i, j) \cdot \mathbf{v}^{(i, j)} = \sum_{1 \leq i, k, j \leq n} \mathbf{M}_u^{(A)}(i, k) \cdot \mathbf{M}_t^{(A)}(k, j) \cdot \mathbf{v}^{(i, j)}$$

Then, for  $w \in \text{WNW}(\Sigma)$  we have

$$\begin{aligned} \mathbf{v}_{ut}(w) &= \sum_{1 \leq i, k, j \leq n} \mathbf{M}_u^{(A)}(i, k) \cdot \mathbf{M}_t^{(A)}(k, j) \cdot \mathbf{v}^{(i, j)}(w) \\ &= \sum_{1 \leq i, k, j \leq n} \mathbf{M}_u^{(A)}(i, k) \cdot \mathbf{M}_t^{(A)}(k, j) \cdot \left( \boldsymbol{\alpha}^T \cdot \mathbf{A}^{(i, j)} \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta} \right) \end{aligned}$$

Note that the row  $i$  of  $\mathbf{A}^{(i, j)} \mathbf{M}_w^{(A)}$  is row  $j$  of  $\mathbf{M}_w^{(A)}$  and all other rows are zero. Then

$$\begin{aligned} \mathbf{v}_{ut}(w) &= \sum_{1 \leq i, k, j \leq n} \mathbf{M}_u^{(A)}(i, k) \cdot \mathbf{M}_t^{(A)}(k, j) \cdot \left( \sum_{l=1}^n \boldsymbol{\alpha}(i) \cdot \mathbf{M}_w^{(A)}(j, l) \cdot \boldsymbol{\eta}(l) \right) \\ &= \sum_{1 \leq i, k, j, l \leq n} \boldsymbol{\alpha}(i) \cdot \mathbf{M}_u^{(A)}(i, k) \cdot \mathbf{M}_t^{(A)}(k, j) \cdot \mathbf{M}_w^{(A)}(j, l) \cdot \boldsymbol{\eta}(l) \\ &= \boldsymbol{\alpha}^T \cdot \mathbf{M}_u^{(A)} \cdot \mathbf{M}_t^{(A)} \cdot \mathbf{M}_w^{(A)} \cdot \boldsymbol{\eta} = \mathbf{r}_{ut}(w) \end{aligned}$$

□

From Theorem 12 and Theorem 5 we have our main result, Theorem 2.

## Acknowledgments.

We thank Boaz Blankrot for helpful discussions on matrix decompositions and the anonymous referees for valuable feedback.

## References

- [1] C. Allauzen, M. Mohri, and M. Riley. Statistical modeling for unit selection in speech synthesis. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 55. Association for Computational Linguistics, 2004.
- [2] R. Alur, M. Arenas, P. Barceló, K. Etessami, N. Immerman, and L. Libkin. First-order and temporal logics for nested words. In *Logic in Computer Science, 2007. LICS 2007. 22nd Annual IEEE Symposium on*, pages 151–160. IEEE, 2007.
- [3] R. Alur and P. Madhusudan. Adding nesting structure to words. In *Developments in Language Theory*, pages 1–13. Springer, 2006.
- [4] D. Angluin. On the complexity of minimum inference of regular sets. *Information and Control*, 39(3):337–350, 1978.
- [5] D. Angluin. Learning regular sets from queries and counterexamples. *Information and computation*, 75(2):87–106, 1987.

- [6] A. Arnold and J. Plaice. *Finite transition systems: semantics of communicating systems*. Prentice Hall International (UK) Ltd., 1994.
- [7] B. Balle and M. Mohri. Spectral learning of general weighted automata via constrained matrix completion. In *Advances in neural information processing systems*, pages 2168–2176, 2012.
- [8] B. Balle and M. Mohri. Learning weighted automata. In *Algebraic Informatics*, pages 1–21. Springer, 2015.
- [9] A. Beimel, F. Bergadano, N. Bshouty, E. Kushilevitz, and S. Varricchio. Learning functions represented as multiplicity automata. *Journal of the ACM (JACM)*, 47(3):506–530, 2000.
- [10] F. Bergadano and S. Varricchio. Learning behaviors of automata from multiplicity and equivalence queries. *SIAM Journal on Computing*, 25(6):1268–1280, 1996.
- [11] L. Bisht, N. Bshouty, and H. Mazzawi. *On optimal learning algorithms for multiplicity automata*. Springer, 2006.
- [12] J. Carlyle and A. Paz. Realizations by stochastic finite automata. *J. Comp. Syst. Sc.*, 5:26–40, 1971.
- [13] K. Chatterjee, L. Doyen, and T. Henzinger. Probabilistic weighted automata. In *CONCUR 2009-Concurrency Theory*, pages 244–258. Springer, 2009.
- [14] K. Chatterjee, T. Henzinger, B. Jobstmann, and R. Singh. Measuring and synthesizing systems in probabilistic environments. In *Computer Aided Verification*, pages 380–395. Springer, 2010.
- [15] A. Cobham. Representation of a word function as the sum of two functions. *Mathematical Systems Theory*, 11:373–377, 1978.
- [16] B. Courcelle and J. Engelfriet. *Graph structure and monadic second-order logic: a language-theoretic approach*, volume 138. Cambridge University Press, 2012.
- [17] B. Courcelle, J. Makowsky, and U. Rotics. Linear time solvable optimization problems on graph of bounded clique width, extended abstract. In J. Hromkovic and O. Sykora, editors, *Graph Theoretic Concepts in Computer Science, 24th International Workshop, WG’98*, volume 1517 of *Lecture Notes in Computer Science*, pages 1–16. Springer Verlag, 1998.
- [18] B. Courcelle, J. Makowsky, and U. Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second order logic. *Discrete Applied Mathematics*, 108(1-2):23–52, 2001.
- [19] K. Culik II and J. Kari. Image compression using weighted finite automata. In *Mathematical Foundations of Computer Science 1993*, pages 392–402. Springer, 1993.
- [20] L. D’Antoni and R. Alur. Symbolic visibly pushdown automata. In *Computer Aided Verification*, pages 209–225. Springer, 2014.
- [21] B. De Schutter and B. De Moor. The singular-value decomposition in the extended max algebra. *Linear Algebra and Its Applications*, 250:143–176, 1997.
- [22] B. De Schutter and B. De Moor. The QR decomposition and the singular value decomposition in the symmetrized max-plus algebra revisited. *SIAM review*, 44(3):417–454, 2002.

- [23] R. Downey and M. Fellows. *Parametrized Complexity*. Springer, 1999.
- [24] E. Driscoll, A. Burton, and T. Reps. Checking compatibility of a producer and a consumer. Citeseer, 2011.
- [25] E. Driscoll, A. Thakur, and T. Reps. Opennwa: A nested-word automaton library. In *Computer Aided Verification*, pages 665–671. Springer, 2012.
- [26] M. Droste and P. Gastin. Weighted automata and weighted logics. In *ICALP 2005*, pages 513–525, 2005.
- [27] M. Droste, W. Kuich, and H. Vogler. *Handbook of weighted automata*. Springer Science & Business Media, 2009.
- [28] C. Fernando, N. Pereira, and M. Riley. Speech recognition by composition of weighted finite automata. *Finite-State Language Processing. MIT Press, Cambridge, Massachusetts*, 1997.
- [29] M. Fliess. Matrices de hankel. *J. Math. Pures Appl*, 53(9):197–222, 1974.
- [30] M. Freedman, L. Lovász, and A. Schrijver. Reflection positivity, rank connectivity, and homomorphism of graphs. *Journal of the American Mathematical Society*, 20(1):37–51, 2007.
- [31] O. Gauwin and J. Niehren. Streamable fragments of forward xpath. In *Implementation and Application of Automata*, pages 3–15. Springer, 2011.
- [32] J. Gentle. *Computational statistics*, volume 308. Springer, 2009.
- [33] B. Godlin, T. Kotek, and J. Makowsky. Evaluation of graph polynomials. In *34th International Workshop on Graph-Theoretic Concepts in Computer Science, WG08*, volume 5344 of *Lecture Notes in Computer Science*, pages 183–194, 2008.
- [34] E. Gold. Complexity of automaton identification from given data. *Information and control*, 37(3):302–320, 1978.
- [35] G. Golub and C. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.
- [36] A. Habrard and J. Oncina. Learning multiplicity tree automata. In *Grammatical Inference: Algorithms and Applications*, pages 268–280. Springer, 2006.
- [37] W. R. Harris, S. Jha, and T. Reps. Secure programming via visibly pushdown safety games. In *Computer Aided Verification*, pages 581–598. Springer, 2012.
- [38] D. Haussler, N. Littlestone, and M. Warmuth. Predicting  $\{0, 1\}$ -functions on randomly drawn points. In *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 100–109. IEEE, 1988.
- [39] A. Heller. Probabilistic automata and stochastic transformations. *Theory of Computing Systems*, 1(3):197–208, 1967.
- [40] D. Hsu, S. Kakade, and T. Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.
- [41] S. Kiefer, A. S. Murawski, J. Ouaknine, B. Wachter, and J. Worrell. On the complexity of equivalence and minimisation for Q-weighted automata. *Logical Methods in Computer Science (LMCS)*, 9(1:8):1–22, 2013.

- [42] V. Klema and A. Laub. The singular value decomposition: Its computation and some applications. *Automatic Control, IEEE Transactions on*, 25(2):164–176, 1980.
- [43] N. Labai. Definability and hankel matrices. Master’s thesis, Technion - Israel Institute of Technology, Faculty of Computer Science, 2015.
- [44] N. Labai and J. Makowsky. Weighted automata and monadic second order logic. *EPTCS Proceedings of GandALF*, 119:122–135, 2013.
- [45] N. Labai and J. Makowsky. Tropical graph parameters. *DMTCS Proceedings of FPSAC*, (01):357–368, 2014.
- [46] N. Labai and J. Makowsky. Meta-theorems using hankel matrices. 2015.
- [47] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2(4):285–318, 1988.
- [48] L. Lovász. Connection matrices. *OXFORD LECTURE series IN MATHEMATICS AND ITS APPLICATIONS*, 34:179, 2007.
- [49] L. Lovász. *Large Networks and Graph Limits*, volume 60 of *Colloquium Publications*. AMS, 2012.
- [50] J. Makowsky. Algorithmic uses of the Feferman-Vaught theorem. *Annals of Pure and Applied Logic*, 126.1-3:159–213, 2004.
- [51] C. Mathissen. Weighted logics for nested words and algebraic formal power series. In *Automata, Languages and Programming*, pages 221–232. Springer, 2008.
- [52] K. McMillan. *Symbolic model checking*. Springer, 1993.
- [53] M. Mohri. Finite-state transducers in language and speech processing. *Computational linguistics*, 23(2):269–311, 1997.
- [54] B. Mozafari, K. Zeng, and C. Zaniolo. High-performance complex event processing over xml streams. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 253–264. ACM, 2012.
- [55] A. S. Murawski and I. Walukiewicz. Third-order idealized algol with iteration is decidable. In *Foundations of Software Science and Computational Structures*, pages 202–218. Springer, 2005.
- [56] L. Pitt and M. Warmuth. The minimum consistent dfa problem cannot be approximated within any polynomial. *Journal of the ACM (JACM)*, 40(1):95–142, 1993.
- [57] A. Poularikas. *Transforms and applications handbook*. CRC press, 2010.
- [58] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.